Лабораторна робота №1

Тема: "Процеси та потоки"

Генерація випадкових чисел та запис їх в файл data.dat

Лістинг:

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <ctime>

using namespace std;

int main() {
    // Відкриваємо файл для запису
    ofstream file("data.dat");

    // Перевіряємо, чи файл вдалося відкрити
    if (!file.is_open()) {
        cout << "Помилка відкриття файлу для запису!" << endl;
        return 1;
    }

    // Встановлюємо початок генератора випадкових чисел залежно від поточного
часу
    srand(time(0));

    // Генеруємо випадкову кількість чисел в діапазоні від 20 до 30
    int count = rand() % 11 + 20;

    // Генеруємо та записуємо випадкові числа в файл
    for (int i = 0; i < count; ++i) {
        // Генеруємо випадкове число в діапазоні від 10 до 100
        int number = rand() % 91 + 10;

        // Записуємо число у файл
        file << number << " ";
    }

    // Закриваємо файл
    file.close();

    cout << "Файл успішно заповнено випадковими числами." << endl;

    return 0;
}
```

Опис: Ця програма написана на мові програмування C++ і має на меті створення файлу з випадково згенерованими числами у заданому діапазоні.

Глобальні змінні:

- Немає глобальних змінних у цій програмі.

Методи та прийоми:

1. Використання функцій з бібліотеки `fstream` для роботи з файлами.
2. Використання функції `rand()` для генерації випадкових чисел.
3. Використання функції `srand(time(0))` для ініціалізації генератора випадкових чисел.

4. Використання циклу `for` для генерації та запису випадкових чисел у файл.
5. Використання функції `close()` для закриття файлу після закінчення запису.

Структура програми:

1. Підключення необхідних бібліотек.
2. Оголошення функції `main()`.
3. Внутрішній блок функції `main()`:
    - Відкриття файлу для запису.
    - Ініціалізація генератора випадкових чисел.
    - Генерація випадкової кількості чисел у заданому діапазоні.
    - Запис випадкових чисел у файл.
    - Закриття файлу.
4. Повернення значення 0.

Сортування файлу data.dat

Лістинг:

```
#include <iostream>
#include <fstream>
#include <vector>
#include <algorithm>
#include <chrono>
#include <thread>
#include <mutex> // для м'ютекса

using namespace std;

void sortArray(vector<int>& arr) {
    // Сортування масиву
    sort(arr.begin(), arr.end());
}

int main() {
    ifstream file("data.dat");
    if (!file.is_open()) {
        cout << "Помилка відкриття файлу!" << endl;
        return 1;
    }

    vector<int> data;
    int temp;
    while (file >> temp) {
        data.push_back(temp);
    }
    file.close();

    // Створення м'ютекса
    mutex mtx;

    // Очікування на натискання клавіші "пробіл" для початку сортування
    cout << "Натисніть пробіл, щоб почати сортування: ";
    char key;
    cin >> key;

    if (key != ' ') {
        cout << "Не вірно натиснута клавіша!" << endl;
        return 1;
    }

    // Блокування м'ютекса перед доступом до спільних даних
```

```
    mtx.lock();

    // Сортування масиву
    sortArray(data);

    // Розблокування м'ютекса після завершення доступу до спільних даних
    mtx.unlock();

    // Оновлення файлу з відсортованим масивом
    ofstream outputFile("data.dat"); // Змінено назву файлу на data.dat
    if (!outputFile.is_open()) {
        cout << "Помилка відкриття файлу для запису!" << endl;
        return 1;
    }

    for (int num : data) {
        outputFile << num << " ";
    }
    outputFile.close();

    cout << "Робота завершена. Відсортовані дані збережено у файлі 'data.dat'."
<< endl;

    return 0;
}
```

Ця програма на мові C++ призначена для сортування чисел, зчитаних з файлу `data.dat`, та запису відсортованих чисел назад у той же файл `data.dat`.

Опис коду:

1. Підключаються необхідні бібліотеки:
    - `<iostream>` для введення/виведення через консоль;
    - `<fstream>` для роботи з файлами;
    - `<vector>` для використання векторів;
    - `<algorithm>` для використання функції сортування;
    - `<chrono>` та `<thread>` для використання затримки;
    - `<mutex>` для роботи з м'ютексами.
2. Оголошується функція `sortArray`, яка приймає посилання на вектор цілих чисел та сортує його за допомогою стандартної функції `sort` з бібліотеки `<algorithm>`.
3. У головній функції `main`:
    - Відкривається файл `data.dat` для зчитування.
    - Числа зчитуються з файлу та зберігаються у векторі `data`.
    - Створюється м'ютекс `mtx` для синхронізації доступу до спільних даних.
    - Очікується натискання клавіші "пробіл" для початку сортування.
    - Перевіряється правильність натискання клавіші "пробіл".
    - Блокується м'ютекс перед сортуванням масиву чисел.
    - Викликається функція `sortArray` для сортування масиву чисел.
    - Розблокується м'ютекс після завершення сортування.
    - Відкривається файл `data.dat` для запису відсортованих чисел.
    - Відсортовані числа записуються назад у файл `data.dat`.
    - Повідомляється про завершення роботи програми.

Виведення даних з фалу data.dat

Лістинг:

```
#include <iostream>
#include <fstream>
#include <chrono>
#include <thread>
```

```
#include <mutex> // для м'ютекса

using namespace std;

int main() {
    ifstream file("data.dat");
    if (!file.is_open()) {
        cout << "Помилка відкриття файлу!" << endl;
        return 1;
    }

    char ch;

    // Створення м'ютекса
    mutex mtx;

    while (file.get(ch)) {
        // Блокування м'ютекса перед доступом до спільних даних
        mtx.lock();

        if (ch == ' ') {
            cout << "*"; // Вивід символу '*' замість числа
        }
        else if (ch != '\n') {
            cout << ch;
        }
        else {
            cout << endl;
        }

        // Розблокування м'ютекса після завершення доступу до спільних даних
        mtx.unlock();

        // Затримка на 0.5 секунди
        this_thread::sleep_for(chrono::milliseconds(500));
    }
    file.close();

    return 0;
}
```

Ця програма на мові C++ призначена для зчитування символів з файлу `data.dat`, виведення їх у консоль з певними обробками та затримкою у виконанні за допомогою м'ютексів.

Опис коду:

1. Підключення необхідних бібліотек:
    - `<iostream>` для введення/виведення через консоль;
    - `<fstream>` для роботи з файлами;
    - `<chrono>` та `<thread>` для роботи зі затримкою;
    - `<mutex>` для роботи з м'ютексами.
2. У головній функції `main`:
    - Відкривається файл `data.dat` для зчитування.
    - Перевіряється, чи відкрився файл успішно. У випадку невдачі виводиться повідомлення про помилку та програма завершується з кодом помилки 1.
    - Створюється м'ютекс `mtx` для синхронізації доступу до спільних даних.
    - Зчитуються символи з файлу `data.dat` у циклі за допомогою функції `file.get(ch)`.
    - Перевіряється кожен зчитаний символ:
        - Якщо символ є пробілом, виводиться символ '*' замість числа.
        - Якщо символ не є символом нового рядка (`\n`), виводиться сам символ.

        - Якщо символ є символом нового рядка, виводиться символ нового рядка.
    - Після кожного виводу символу заблокований м'ютекс розблоковується, щоб інші потоки могли мати доступ до спільних даних.
    - Перед виведенням наступного символу потік затримується на 0.5 секунди.
    - Після завершення зчитування файлу він закривається.
    - Повертається 0, щоб позначити успішне завершення програми.